A Study on Hierarchical Model of a Computer Worm Defense System

Basheer Suleiman, Rashid Husain, Saifullahi Muhammad

***Abstract*— This research addresses the problem of computer worms in the modern Internet. A worm is similar to a virus. A worm is a self-propagating computer program that is being increasingly and widely used to attack the Internet. It is considered as a sub-class of a virus because it is also capable of spreading from one computer to another. Worms are also computer programs that are capable of replicating copies of themselves via network connections. What makes it different however is that unlike a computer virus a computer worm can run itself without any human intervention? Because of these two qualities of a worm, it is possible that there will be thousands of worms in a computer even if only one computer worm is transferred. For instance, the worm may send a copy of itself to every person listed in the e-mail address book. The worm sent may then send a copy of itself to every person who is listed in the address book of the person who receives the email. Because this may go on ad infinitum worms can not only cause damage to a single computer and to other person's computer but it can only affect the functionality of Web servers and network servers to the point that they can no longer function efficiently. One example is the .blaster worm.**

***Index Terms*— Worm, Hierarchical Model, Internet, Hosts, Infection, Protected, Firewall**

I. INTRODUCTION

This model assumes a tree structure where the internal nodes of the tree are firewalls and leaves are servers vulnerable to worm attacks. See Figure 1.1. The firewalls are assumed to be immune to infections [1, 2]. It is also assumed that we have sensors at the vulnerable hosts that can detect an infection and report it. All nodes at a particular level of the hierarchy have equal number of children. Each level of the hierarchy has a certain threshold associated with it. Once the number of infection reports amongst a node's (firewall's) children reaches the threshold, the firewall turns on the filter rules protecting all of its children, and alerts its parent that the sub-tree below it is infected but now protected. This escalation of alerts from one level to the next higher level in the hierarchy and protection of sub-trees takes place successively as the threshold for infections is reached at each node. False alerts are handled by the firewalls by associating a `Time to Live'(TTL) with the latest alert that they receive. If the threshold is reached before the TTL expires the above actions are taken. If the TTL expires before the threshold is hit, all the alerts/infection reports are discarded and the firewalls `back-off' from protecting its children [2, 4]. This is done because any action taken to contain a worm involves a cost, and actions undertaken when there is no worm are of no use.

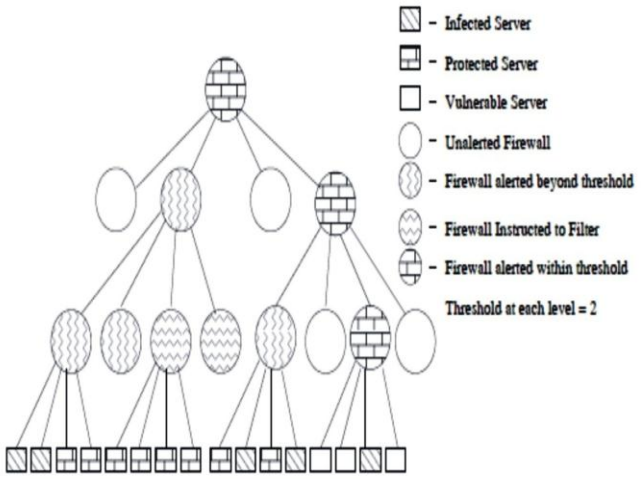

Fig.1.1: Hierarchical relationship among firewalls and vulnerable hosts in Hierarchical Model

II. MATHEMATICAL MODELS

We will derive some of the interesting quantities of the hierarchical model of mitigation mathematically. As always, the goal of this whole project is to minimize the number of infections amongst our participating hosts [4, 7].
The first quantity of interest is the probability with which infections are minimized. For this to happen, the root node should trigger the protection of its children with the minimum number of hosts being infected [3, 4].

Probability to minimize infection For the sake of this derivation let us consider that the leaves are at level 0 of the tree, the parents of the leaves are at level 1 and so on [23, 24].

$$Number\ of\ children\ per\ leel = \lambda$$
$$Number\ of\ levels = n$$
$$Number\ of\ leaes = \lambda^{n-1} = m_0$$
$$Number\ of\ nodes\ one\ leel\ above\ leaves = \lambda^{n-2} = m_1$$
$$P\ (an\ infection\ choosing\ one\ particular\ domain) = {}^{1}/_{\lambda^{n-2}} = {}^{1}/_{m_1}$$
$$P\ (of\ i\ infections\ choosing\ the\ same\ particular\ domain) = {}^{1}/_{\lambda^{n-i}} = {}^{1}/_{m_1^i}$$
$$Number\ of\ ways\ this\ one\ domain\ can\ be\ chosen = C_1^{m_1} = m_1$$
$$Threshold = \tau - 1$$
$$Minimum\ number\ of\ infections\ needed\ at\ the\ leaes\ level\ to\ alert\ the\ root = \tau^{n-1}$$

For a node to protect its children, it should have received τ alerts from its children.
That is the number of infections $i = \tau$.

$$P\ (some\ non-leaf\ node\ at\ level\ 1\ to\ protect\ to\ its\ children) = {}^{C_1^{m_1}}/_{m_1^{\tau}}$$

Basheer Suleiman, Research Scholor,Umaru Musa Yar'aduva University, Nigeria
Rashid Husain, Lecturer, Umaru Musa Yar'aduva University, Nigeria
Saifullahi Muhammad, Lecturer, Federal University, Dusin-Ma, Nigeria

$$= {m_1}/{m_1^{\tau}}$$
$$= {1}/{m_1^{\tau-1}}$$

Similarly,

P (Some non-leaf node at level 2 to protect to its children)

$$= \left(\frac{1}{m_2^{\tau-1}}\right).P(\tau \text{ nodes being alerted at level } 1)$$
$$= \left(\frac{1}{m_2^{\tau-1}}\right).\left(\frac{1}{m_1^{\tau-1}}\right)^{\tau}$$

P (Some non-leaf node at level 3 to protect to its children)

$$= \left(\frac{1}{m_3^{\tau-1}}\right).P(\tau \text{ nodes being alert at level } 2)$$
$$= \left(\frac{1}{m_3^{\tau-1}}\right).\left[\left(\frac{1}{m_2^{\tau-1}}\right).\left(\frac{1}{m_1^{\tau-1}}\right)^{\tau}\right]^{\tau}$$
$$= \left(\frac{1}{m_3^{\tau-1}}\right)\left(\frac{1}{m_2^{\tau-1}}\right)^{\tau}.\left(\frac{1}{m_1^{\tau-1}}\right)^{\tau^2}$$

. . .

P (some non-leaf node at level (n - 1) to protect to its children)

$$= \left(\frac{1}{m_{n-1}^{\tau-1}}\right)^{\tau^0}\left(\frac{1}{m_{n-2}^{\tau-1}}\right)^{\tau^1}\left(\frac{1}{m_{n-3}^{\tau-1}}\right)^{\tau^2}\cdots\left(\frac{1}{m_1^{\tau-1}}\right)^{\tau^{n-2}}$$
$$= \prod_{i=1}^{n-1}\left(\frac{1}{m_{n-i}^{\tau-1}}\right)^{\tau^{i-1}}$$

But the only node at level n − 1 is the root node. So, the above expression gives us the probability that the root will be alerted with the minimum number of infections at the leaves.

Time to alert root The next interesting mathematical result is the time it takes to achieve complete protection. That is the time it takes to alert the root so that it can trigger protection of its children [17, 18].

For the sake of this derivation we will consider a small sub-tree with just 2 levels. The top level contains the root node and the leaves all form a group. Each infected leaf node tries to infect all non-infected leaf nodes in its group [8, 9]. For each infected/non-infected leaf node pair, the time until infection of the non-infected node by the infected one is exponentially distributed. By scaling of time, we can take the mean of this distribution to be 1.0. It is assumed that all infecting processes operate stochastically independently [6, 12].

i: Number of infected nodes in a group

h: The threshold for alerts

g: The group size

Then in state i, there are $i(g-i)$ exponential random variables in progress at once, since each of the i infected nodes is trying to infect each of the g−i uninfected nodes. Then the time to go from state i to $i+1$ will be the minimum of $i(g-i)$ exponentially distributed random variables, and thus will itself be exponentially distributed1, with mean $1.0/[i(g-i)]$ [7,8].

For simplicity, we will consider the case $h = g-1$; the more general case is handled similarly. The total expected time to an alert, starting at the time the first member of the group becomes infected, is [10, 24]

$$\sum_{i=1}^{g-1}\frac{1}{i(g-1)} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots(1.1)$$

Using a standard approximation, (1.1) is approximately equal to

$$\int_1^{g-1}\frac{1}{x(g-x)}\,dx = \frac{1}{g}\int_1^{g-1}\left(\frac{1}{x}+\frac{1}{g-x}\right)dx = \frac{2}{g}\ln(g-1)+C \qquad \ldots\ldots\ldots\ldots\ldots(1.2)$$

Where C is the constant of integration. The latter quantity goes to C as $g \to \infty$.

In other words, (1.1) remains bounded as $g \to \infty$. This is a very interesting result, since it says that the mean time to alert is bounded no matter how big our group size is [14, 15].

This is verified in our simulations.

III. Description of the Simulation

The Hierarchical Model simulation was implemented in Perl. The simulation was done on a network modeled as a tree with 4 levels. Each level of the tree has 4 children. The simulation is started by randomly infecting a single leaf node. The rate of infection is fixed at the beginning of the simulation. The exact number of machines that each infected machine tries to infect is determined by using a Poisson distribution, with the mean value as the rate of infection. The worm scenarios were simulated with a large TTL value, 1000 [17, 21].

In each time slice, every infected machine tries to infect as many other machines as dictated by the Poisson distribution. Alerts are raised in the same time slice as an infection occurs. And each alert is propagated as high as possible in the hierarchy in the same time slice.

In the case of false alarms, the rate of false alarms is fixed at the beginning of the simulation. Unlike worms where only infected machines can infect others, there is no relationship between one false alarm and the next. The machine that raises a false alarm is determined randomly [19, 21]. All false alarm scenarios were simulated with TTL window sizes varying from 10 to 400 time units.

Simulations were run with thresholds at 75% and 50% of the number of children. That is, if 75% of a node's children have raised alerts, the node takes action. The structure of network and thresholds were chosen so as to be comprehensible. However, more complex structures with different number of children at each level and different thresholds at each level could also be simulated [24].

IV. Discussion of the Results

The basic results of two extreme cases where all parameters are identical except the rate of infection which is very high and very low are shown in Fig.1.2 and Fig.1.3. These two figures show that the number of infections before complete immunization could take place is almost the same for both the cases [22].

The simulation was done for different rates of infection with 2 different levels of thresholds and the results are shown in Fig.1.4. As we can see in Fig.1.4, the number of infections before complete immunization takes place is almost the same for varied rates of infection. But the time it takes is different

and favorable too as seen in Fig.1.5. For high rates of infection, maximum possible protection is achieved quicker than for slower rates of spread [23]. This is a direct result of the dependence of the alert propagation on the infection rate. And we also see that the thresholds don't determine the time taken as it takes almost the same time to achieve complete immunization for 2 different threshold values. The only thing that varies with threshold is the number of infected machines [19, 20].

Thus the lessons learnt are [11, 13]:

- For any rate of infection, the number of victims is the same.

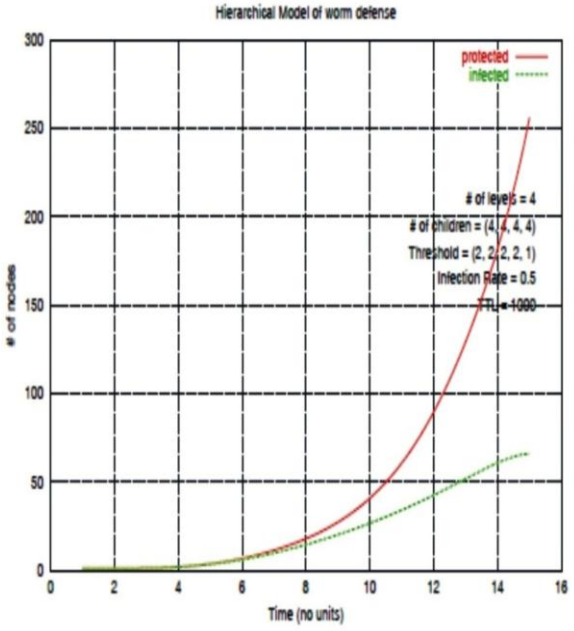

Figure 1.2: Response for a low rate of infection. # represents the number

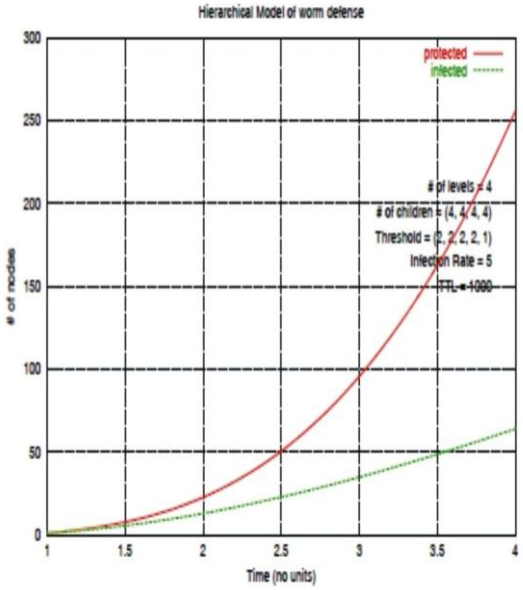

Figure 1.3: Response for a high rate of infection. # represents the number

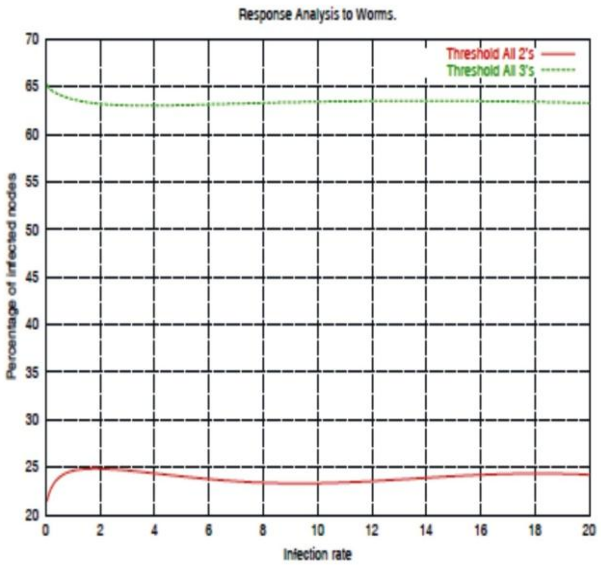

Figure 1.4: Percentage of machines infected for different rates of infection.

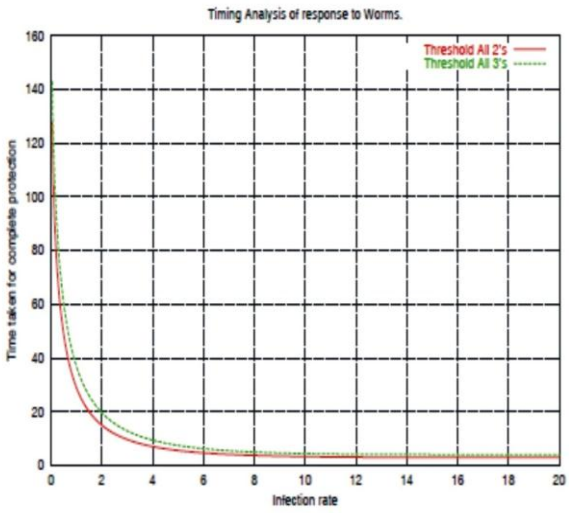

Figure 1.5: The time taken for complete protection is inversely proportional to the infection rate.

- The time taken for complete protection is inversely proportional to the infection rate.
- It is only the threshold levels that makes or breaks the network. A low threshold helps to save a lot of machines.

It is obvious that we can't forecast an unknown worm's infection rate. But we can define our tolerance. So, we now have a model which can tell us what should be the threshold, for a given tolerance.

For example, if we want to save 90% of the Internet in the event of a worm attack, we need to find out from simulations what should be the thresholds at various levels of the network hierarchy and set the firewall rules accordingly. Setting the right thresholds at various levels of the hierarchy would achieve our goal of saving 90% of the Internet for us slowly or quickly, depending on the infection rate of the worm [21, 22, 23].

4.1 False Alarms

During false alarms the number of machines given protection does not rise steadily as in case of real worms. Rather, the

number of machines protected keeps oscillating as the systems backs off if there are no alerts within the TTL, as seen in Fig.1.6. This oscillation is an indication of a series of false alarms. It can also be considered as an indication of a Stealth Worm spreading very slowly. This is discussed in the next sub-section [1, 3].

Fig.1.7 shows the average number of machines that are given protection in response to false alarms for various TTLs and various false alarm rates. This protection involves a price and can be considered as a self inicted DoS attack [4, 5]. As we can see, if the TTLs are low enough, we pay a much lower price. But we would not be able to capture Stealth worms as is seen in figure 1.9. At the same time, a high TTL would raise false positives even for low rates of false alarms and raise costs unnecessarily.

4.2 Stealth Worms

We can see the same oscillating pattern in figure 1.8 and figure 1.9 which were recorded for a Stealth worm simulation with a TTL of 60. Figure 1.8 shows a case where the stealth worm is suppressed because of a low threshold of the defense system [16, 17].

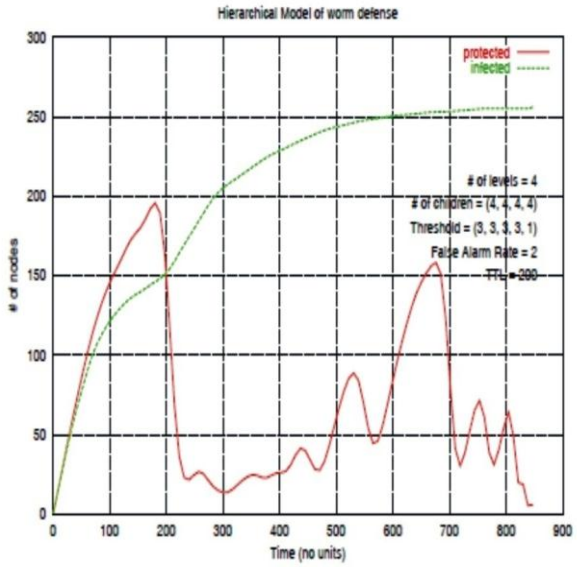

Figure 1.6: The number of machines that are protected keeps oscillating in case of false alarms. # represents the number

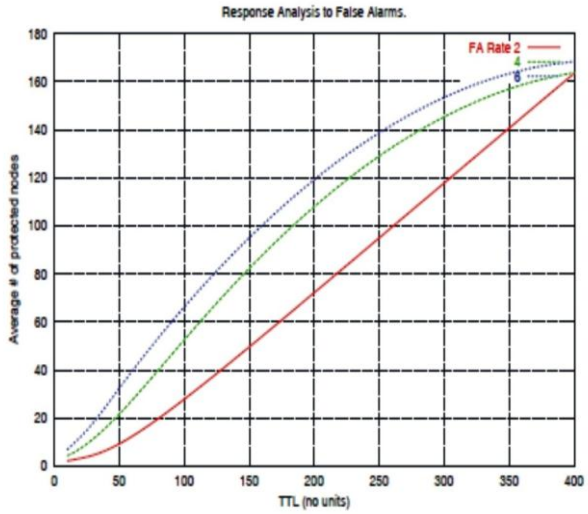

Figure 1.7: Average number of machine protected for different False Alarm rates for varying TTLs.

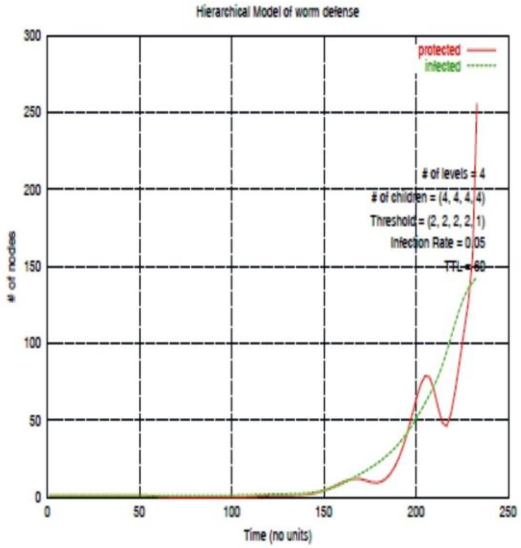

Figure 1.8: A stealth worm overpowered with a low threshold. # represents the number

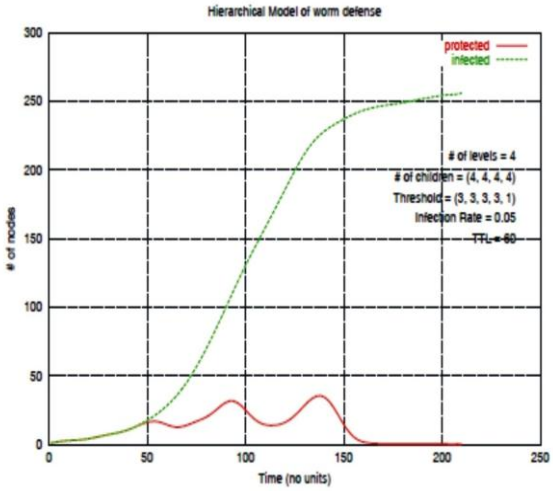

Figure 1.9: A Stealth worm sneaking in due to a high threshold.

Fig.1.9 shows a scenario where the protection mechanism is able to sense that something is wrong. But due to the high threshold, the rate of alerts received is just low enough that the TTL expires frequently and our system backs-off reasoning it as a false alarm. This suggests that we need a higher TTL or a lower threshold, which we addressed above. But a high TTL means a high cost due to false alarms which may be unacceptable [23, 24].

Since an oscillating curve means a series of false alarms or a stealth worm which is slow spreading, we can afford the luxury of human intervention.

So, we need to arrive at a compromise saying that we will look out for Stealth worms which only spread above a certain speed. In any case, slower worms will get exposed as more and more machines are infected, as this will increase the overall rate of infection. So, the TTL also dictates how many machines we will have to sacrifice before our defense mechanism takes over. We may even lose all machines before we respond if we choose a very small TTL [16, 19].

V. CONCLUSIONS

This model uses a hierarchical relationship between co-operating hosts to mitigate the spread of a worm. Each leaf node is a vulnerable host and each non-leaf node is an invulnerable control structure which can process alerts sent by sensors at leaves and each non-leaf can issue appropriate instructions to its children.

This paper also provided mathematical analyses of the hierarchical model of worm defense and showed that the time to alert the root of the hierarchy is bounded.

From this model, and the simulations, we can determine the thresholds required at various levels of the hierarchy for a given tolerance of lost machines. We can also determine the TTL to stop a Stealth worm of a given speed by looking at the usual rate of false alarms in an environment. With the threshold levels and TTLs thus determined, we can effectively inhibit the spread of worms without losing much due to false alarms.

REFERENCES

[1] C.G.Senthilkumar, (2002). "Worms: How to stop them? - A proposal for Master's thesis," *University of California, Davis. http://wwwcsif.cs.ucdavis*.

[2] C.G.Senthilkumar and Karl Levitt, (2003). "\Hierarchically Controlled Co-operative Response Strategies for Internet Scale Attacks," *University of California, Davis. http://wwwcsif.cs.ucdavis.edu*.

[3] D.Farmer and W.Venema, (2004). "Security Administrator's tool for analyzing networks" *http://www._sh.com//zen/satan/satan.html*.

[4] D.Noijiri, J.Rowe, and K.Levitt, (2002). "Cooperative Response Strategies for Large Sacle Attack Mitigation," *DISCEX*.

[5] Mark W. Eichin and Jon A. Rochlis, (1988). "With Microscope and Tweezers: An analysis of the Internet Virus of November 1988," *In Proceedings of the symposium on Research in Security and Privacy, Oakland, CA*.

[6] Dan Farmer and Eugene H. Spa_ord, (1990). "The cops Security Checker System ," *USENIX*.

[7] Gene Kim and Eugene H. Spa_ord, (1993). "The design of a system integrity monitor: Tripwire," *Technical Report CSD-TR-93-071, Purdue University, West Lafayette, IN, USA*.

[8] David Moore et al. (2003). "Inside the Slammer Worm," *In IEEE Security and Privacy*.

[9] Carey Nachenberg, "Computer Parasitology," *Symantec AntiVirus Research Center*.

[10] Carey Nachenberg. "Understanding and Managing Polymorphic Viruses," *Symantec AntiVirus Research Center*.

[11] Don Seeley, (1989). "A Tour of the Worm," *In Proceedings of 1989 Winter USENIX Conference*, pp. 287 -304.

[12] John F. Shoch and Jon A. Hupp, (1982). "The Worm Programs - Early Experience with a Distributed Computation," *Communications of the ACM*, Vol.25(3) pp: 172 -180.

[13] Eugene H. Spa_ord, (1988). "The Internet Worm Program: An Analysis," *Technical Report CSD-TR-823, Purdue University, West Lafayette, IN, USA*.

[14] Eugene H. Spa_ord, (1989). "The Internet Worm: Crisis and aftermath," *Communications of the ACM*, Vol. 32(6), pp: 678 – 687.

[15] Eugene H. Spa_ord, (1991). "The Internet Worm Incident," *Technical Report CSD-TR-933, Purdue University, West Lafayette, IN, USA*.

[16] S.Staniford-Chen et al. (1996). "A Graph-Based Intrusion Detection System for Large Networks," *In The 19th National Information Systems Security Conference*, Volume 2, pages 361 – 370.

[17] Stuart Staniford, Gary Grim, and Roelof Jonkman, (2001). "Flash Worms: Thirty Seconds to Infect the Internet," *Silion Defense - Security Information*.

[18] Stuart Staniford, Vern Paxson, and Nicholas Weaver (2002). "How to Own the Internet in Your Spare Time," *In Proceedings of USENIX Conference, Berkeley, Usenix Association, USENIX*.

[19] Nicholas Weaver, (2002). "Future Defenses: Technologies to Stop the Unknown Attack Internet", *http://online.securityfocus.com/infocus/1547*.

[20] Nicholas Weaver, (2002). "Potential Strategies for High Speed Active Worms: A Worst Case Analysis," *UC Berkeley*.

[21] Nicholas Weaver, (2002). "Warhol Worms: The Potential for Very Fast Internet Plagues," *UC Berkeley*.

[22] Tarkan Yetiser, (1993). "Polymorphic Viruses Implementation, Detection and Protection," *VDS Advanced Research Group*.

[23] Dan Zerkle and Karl Levitt, (1996). "Netkuang - a multi-host con_guration vulnerability checker," *USENIX*.

[24] Husain Rashid and Mansir Abubakar, (2015). "A Study on Friends Model of a Computer Worm Defense System", *IJEAS*, Vol. 2(3), pp: 56-59.

.